Shell Script to Generate Daily/Weekly AWR reports (Email)

oracledbasupport.co.uk/shell-script-to-generate-dailyweekly-awr-reports

Create .run_awr with following details :

```
"TNS-connect-string : recipient-list : hrs of AWR snapshot"
[oracle@ ~]$ cat .run_awr
prod:root@oracledbasupport.co.uk:11
I added this script in my crontab for a daily emails:
```

```
########## Daily Export of AWR reports
02 18 * * * /home/oracle/.awr_daily.sh  >> /home/oracle/awr.log 2>&1
```

```
[oracle@awr_reports]$ ls -lrt
-rw-r--r-- 1 oracle oracle 315104 Oct 26 10:02 AWR_26102010_1002_prod.HTML
-rw-r--r-- 1 oracle oracle 343839 Oct 26 18:02 AWR_26102010_1802_prod.HTML
-rw-r--r-- 1 oracle oracle 342611 Oct 27 18:02 AWR_27102010_1802_prod.HTML
-rw-r--r-- 1 oracle oracle 282057 Oct 28 18:02 AWR_28102010_1802_prod.HTML
```

1. **Create AWR report between sysdate and sysdate – hours (download)**

```
#     The file ".run_awr" in the "$HOME" directory contains one or more
#     lines with the following format, three fields delimited by "semicolon":
#
#           TNS-connect-string : recipient-list : hrs
```

2. **Create AWR report between sysdate-days and sysdate – hours (download)**

```
#     The file ".run_awr" in the "$HOME" directory contains one or more
#     lines with the following format, three fields delimited by "semicolon":
#
#           TNS-connect-string : recipient-list : daysInPast : hrs
```

```
#!/usr/bin/ksh

 #===============================================================================

# File:         run_awr.sh
# Type:         korn shell script
#
# Description:
#       UNIX Korn-shell script to run under the UNIX "cron" utility to
#       automatically generate and email Oracle "AWR" reports in HTML against
#       the database accessed via the specified TNS connect-string, to a
#       specified list of email addresses.
#
# Parameters:
#       Zero, one, or more parameters may be passed.  These parameters
#       are TNS connect-strings, each of which refer to entries in the
#       script's configuration file (named ".run_awr", described below).
#
```

```
#       If no parameters are specified, then the script processes all of
#       the lines in the configuration file.
#
#       For each of the parameters specified, the script will process
#       each of the corresponding lines in the configuration file.
#
#       Each TNS connect-string should be separated by whitespace.
#
# Configuration file:
#       The file ".run_awr" in the "$HOME" directory contains one or more
#       lines with the following format, three fields delimited by "commas":
#
#               TNS-connect-string : recipient-list : hrs
#
#       where:
#
#               TNS-connect-string      Oracle TNS connect-string for the db
#               recipient-list          comma-separated list of email addresses
#               hrs                     "sysdate - <hrs>" is the beginning
#                                       time of the AWR report and "sysdate"
#                                       is the ending time of the AWR report
#
# Modification history:
#===============================================================================
#
#-------------------------------------------------------------------------------
# Set up Oracle environment variables...
#-------------------------------------------------------------------------------
export ORACLE_SID=prod
export ORAENV_ASK=NO
. /usr/local/bin/oraenv > /dev/null 2>&1
unset ORAENV_ASK
#
#-------------------------------------------------------------------------------
# Verify that the Oracle environment variables and directories are set up...
#-------------------------------------------------------------------------------
if [[ "${ORACLE_HOME}" = "" ]]
then
 echo "ORACLE_HOME not set; aborting..."
 exit 1
fi
if [ ! -d ${ORACLE_HOME} ]
then
 echo "Directory \"${ORACLE_HOME}\" not found; aborting..."
 exit 1
fi
if [ ! -d ${ORACLE_HOME}/bin ]
then
 echo "Directory \"${ORACLE_HOME}/bin\" not found; aborting..."
 exit 1
fi
if [ ! -x ${ORACLE_HOME}/bin/sqlplus ]
then
 echo "Executable \"${ORACLE_HOME}/bin/sqlplus\" not found; aborting..."
 exit 1
fi
if [ ! -x ${ORACLE_HOME}/bin/tnsping ]
```

```
then
 echo "Executable \"${ORACLE_HOME}/bin/tnsping\" not found; aborting..."
 exit 1
fi
#
#------------------------------------------------------------------------------
# Set shell variables used by the shell script...
#------------------------------------------------------------------------------
_Pgm=AWR_`date '+%d%m%Y_%H%M'`
_RunAwrListFile=${HOME}/.run_awr
if [ ! -r ${_RunAwrListFile} ]
then
 echo "Script configuration file \"${_RunAwrListFile}\" not found;
aborting..."
 exit 1
fi
#
#------------------------------------------------------------------------------
# ...loop through the list of database instances specified in the ".run_awr"
# list file...
#
# Entries in this file have the format:
#
#       dbname:rcpt-list:hrs
#
# where:
#       dbname          - is the TNS connect-string of the database instance
#       rcpt-list       - is a comma-separated list of email addresses
#       hrs             - is the number of hours (from the present time)
#                         marking the starting point of the AWR report
#------------------------------------------------------------------------------
grep -v "^#" ${_RunAwrListFile} | awk -F: '{print $1" "$2" "$3}' | \
while read _ListDb _ListRcpts _ListHrs
do
 #----------------------------------------------------------------------
 # If command-line parameters were specified for this script, then they
 # must be a list of databases...
 #----------------------------------------------------------------------
 if (( $# > 0 ))
 then
 #
 #--------------------------------------------------------------
 # If a list of databases was specified on the command-line of
 # this script, then find that database's entry in the ".run_awr"
 # configuration file and retrieve the list of email recipients
 # as well as the #-hrs for the AWR report...
 #--------------------------------------------------------------
 _Db=""
 _Rcpts=""
 _Hrs=""
 for _SpecifiedDb in $*
 do
 #
 if [[ "${_ListDb}" = "${_SpecifiedDb}" ]]
 then
 _Db=${_ListDb}
 _Rcpts=${_ListRcpts}
 _Hrs=${_ListHrs}
 fi
 #
 done
 #
 #---------------------------------------------------------------
 # if the listed DB is not specified on the command-line, then
 # go onto the next listed DB...
 #---------------------------------------------------------------
 if [[ "${_Db}" = "" ]]
 then
 continue
 fi
 #---------------------------------------------------------------
 else    # ...else, if no command-line parameters were specified, then
 # just use the information in the ".run_awr" configuration file...
 #---------------------------------------------------------------
 _Db=${_ListDb}
 _Rcpts=${_ListRcpts}
 _Hrs=${_ListHrs}
 #
 fi
 #
 #----------------------------------------------------------------------
 # Verify that the name of the database is a valid TNS connect-string...
 #----------------------------------------------------------------------
 ${ORACLE_HOME}/bin/tnsping ${_Db} > /dev/null 2>&1
 if (( $? != 0 ))
 then
 echo "\"tnsping ${_Db}\" failed; aborting..."
 exit 1
 fi
 #
 #----------------------------------------------------------------------
 # Create script variables for the output files...
 #----------------------------------------------------------------------
 _TmpSpoolFile="/home/oracle/awr_reports/${_Pgm}_${_Db}.HTML"
 _AwrReportFile="${_Pgm}_${_Db}.html"
 #
 #----------------------------------------------------------------------
 # Call SQL*Plus, retrieve some database instance information, and then
 # call the AWR report as specified...
 #----------------------------------------------------------------------
 ${ORACLE_HOME}/bin/sqlplus -s /nolog << __EOF__ > /dev/null 2>&1
set echo off feedback off timing off pagesize 0 linesize 300 trimspool on
verify off heading off
connect / as sysdba

col dbid new_value V_DBID noprint
select  dbid from v\$database;

col instance_number new_value V_INST noprint
select  instance_number from v\$instance;

col snap_id new_value V_BID
select  min(snap_id) snap_id
```

```
from    dba_hist_snapshot
where   end_interval_time >= (sysdate-(${_Hrs}/24))
and     startup_time <= begin_interval_time
and     dbid = &&V_DBID
and     instance_number = &&V_INST;

col snap_id new_value V_EID
select  max(snap_id) snap_id
from    dba_hist_snapshot
where   dbid = &&V_DBID
and     instance_number = &&V_INST;

spool ${_TmpSpoolFile}
select  'BEGIN='||trim(to_char(begin_interval_time, 'HH24:MI')) snap_time
from    dba_hist_snapshot
where   dbid = &&V_DBID
and     instance_number = &&V_INST
and     snap_id = &&V_BID ;
select  'END='||trim(to_char(end_interval_time, 'HH24:MI')) snap_time
from    dba_hist_snapshot
where   dbid = &&V_DBID
and     instance_number = &&V_INST
and     snap_id = &&V_EID ;
spool off

select output from table(dbms_workload_repository.awr_report_html(&&V_DBID,
&&V_INST, &&V_BID, &&V_EID, 0))

spool /tmp/${_AwrReportFile}
/
exit success
__EOF__
 #
 #----------------------------------------------------------------------
 # Determine if the "start time" and "end time" of the AWR report was
 # spooled out...
 #----------------------------------------------------------------------
 if [ -f ${_TmpSpoolFile} ]
 then
 _BTstamp=`grep '^BEGIN=' ${_TmpSpoolFile} | awk -F= '{print
$2}'`
 _ETstamp=`grep '^END=' ${_TmpSpoolFile} | awk -F= '{print $2}'`
 fi
 #
 #----------------------------------------------------------------------
 # Determine if an AWR report was spooled out...
 #----------------------------------------------------------------------
#                if [ -f /tmp/${_AwrReportFile} ]
#                then
#
#                        uuencode /tmp/${_AwrReportFile} ${_AwrReportFile} | \
#                                mailx -s "AWR Report for ${_Db}
#        (${_BTstamp}-${_ETstamp} GMT)" ${_Rcpts}
#
#                fi
#
```

```
mv /tmp/${_AwrReportFile} ${_TmpSpoolFile}
done
#
#-------------------------------------------------------------------------------
# Finish up...
#-------------------------------------------------------------------------------
exit 0
```